Python Basics

Ranjan Mondal

Indian Statistical Institute

8th April 2017

Function

- $f : a \rightarrow b$
  - Function f maps from set a to set b
  - $b1 = f(a1)$: Function takes some data $a1$ and produce $b1$
- Multiple return
- No "call-by-reference" nor "call-by-value" concept
- Recursion
  - Three concept: arguments,return,halting conditions
  - Recursion with no arguments and no return
  - Recursion with arguments and no return
  - Recursion with arguments and and return

File Handeling

- A file object is created using open function before you read or write.
  - f=open(file_name,access_mode). This returns a file object if successful else IOError
  - access_mode determines the mode in which the file has to be opened, i.e., read, write, append
    - access_mode "r" : Opens a file for reading only
    - access_mode "w" : Opens a file for writing only. Overwrites the file if the file exists
    - access_mode "w+" : Opens a file for both writing and reading. Overwrites the existing file if the file exists
    - access_mode "a+" : Opens a file for both appending and reading

Read a file

- Step 1 :Open a file in read mode. f=open("file_name","r")
- Step 2 :Read
  - f.read() read the whole file into a string.
  - f.readline() read each line and return string
  - f.readlines() read all the lines and return a list of string with all the lines.
- step 3: f.close()

Write into a file

- Step 1 :Open a file in write mode. f=open("file_name","w")
- Step 2 :write
  - f.write(str) :Write string str to file
  - f.writelines(sequence_of_strings) : Write the strings to the file
- step 3: f.close()

Try out

- write a function that will take a text file name with path and it will print the file line by line .
  - func(file_name)

Python Data Structures

There four basic Data Structures available in Python.

1. Lists

Python Data Structures

There four basic Data Structures available in Python.

1. Lists
2. Tuples

Python Data Structures

There four basic Data Structures available in Python.

1. Lists
2. Tuples
3. Dictionary

Python Data Structures

There four basic Data Structures available in Python.

1. Lists
2. Tuples
3. Dictionary
4. Strings

Python Data Structures

There four basic Data Structures available in Python.

1. Lists
2. Tuples
3. Dictionary
4. Strings
5. Sets

Lists

- An ordered group of items(objects).
- Does not need to be the same data(object) type.
- List items is accessed by index.
- Notation: L=[“HI”,“Midnapore”,123,“CS”,2.21,0.222]
- Indexing
  - Starts from 0 in forward direction.
  - Starts from -1 in backward direction.

Methods of Lists

- List.append(x)
  - L=[3,'b','c',3,4,3]
  - Adds an item to the end of the list
  - let L=[3,'b','c',3] ; L.append(4)

Methods of Lists

- List.count(x)
  - L=[3,'b','c',3,4,3]
  - Returns number of occurrences of x
  - L.count(3) returns 2.

Methods of Lists

- List.extend(L1)
  - L=[3,'b','c',3,4,3]
  - Extend the list by appending all in the given list L1
  - L1=[4,3] ; L.extend(L1) ; print L => [3,'b','c',3,4,3]
  - Note the difference between L.append() and L.extend()

Methods of Lists

- List.insert(i,x)
  - L=[3,'b','c',3,4,3]
  - Inserts an item x at index I
  - L[0]='Fun with Python '; print L => ['Fun with Python','b','c',3,4,3]

Methods of Lists

- List.remove(x)
  - L=[3,'b','c',3,4,3]
  - Remove first occurrence of value x
  - L.remove('Fun with Python') ; print L => ['b','c',3,4,3]
  - Try L.remove(3) ?

Try out now

- Use List as Stack.
- Can you use List as array(1D,2D,3D,..) ?

Tuples

- Similar to the list data structure
- Notation: T=('23',389,'easy')
- Tuples are immutable(but lists are). You can't change tuple T like this T[0]=3 . But in list you can.
- Tuples are faster than Lists. why ?

Methods of Tuples

- T.count() , T.index()
  - Same as Lists

Application of Tuples

- Faster than lists
- Protect the data, which is immutable
- Tuples can be used as keys on dictionaries

Dictionary

- Dictionary is a sequence of items.
- Each item is a pair made of a key and a value
- Access to the list of keys or values independently
- Informally we can consider it as hashtable
- Notation: D = {42:'Hi ', 4:'hello',2:32}
  - print D[42] ?
  - print D[2] ?

Methods of Dictionary

- D.values() Returns list of D's values
  - D = {42:'Hi ', 4:'hello',2:32}
  - D.values() returns ['Hi', 'hello', 32]
- D.keys() Returns list of D's keys
- D.iteritems() an iterator over the *key*, *value* items of D
  - for k,v in D.iteritems():
    print k,v
- D.has_key(k) Returns True if D has a key k, else False.
  - D.has_key('Hi') returns ?
  - D.has_key(4) returns ?
  - D.has_key(2) returns ?
- D.pop(k) Removes specified key 'k' and return the corresponding value for the key.

Strings

- Strings are immutable sequence of characters.
- Notation
  - S ='Hi, I am at Midnapore'
  - S =“Hi, I am at Midnapore”
  - S =' ' 'Hi, I am at Midnapore' ' '
  - S = """Hi, I am at Midnapore"""
  - """ S= My name is “ranjan” """
- Indexing
  - Starts from 0 in forward direction.
  - Starts from -1 in backward direction.

Methods of Strings

- S=“python is fun”
- S.count() Return the number of non-overlapping occurrences of substring sub in string S.
- S.isdigit() Return True if all characters in S are digits else False.
- S.split(sep) Return a list of the words in the string S, using sep as the delimiter string. example: S.split(“ ”)
- S.find(s) Return the lowest index in S where substring s is found. Return -1 on failure.
- S.replace(old,new) Return a copy of string S with all occurrences of substring old replaced by new.
  - S=S.replace('python','CS')
- operators + and * to create new strings.
  - S=“ hello”
  - S=S+“ everybody.”+“ How are you?.”
  - S=S*2
  - print S

Sets

- Sets are constructed from a sequence.
- Cannot have duplicated items.
- Notation and Example
  - a = set([1, 2, 3, 4])
  - b = set([3, 4, 5, 6])
  - c=a|b (Union)
    - c is a Union of a and b. i.e [1, 2, 3, 4, 5, 6]
  - c=a&b (Intersection)
    - c is a Intersection of a and b. i.e [3, 4]
  - $a < b$ returns True is a is a subset of b.
  - $a - b$ is a Difference from set a with b.
  - $a \hat{} b$ is a Symmetric Difference between a and b.

THANK YOU